Subrutine, întreruperi, şi serviciiSubrutineSubrutina este o secvenţă de instrucţiuni scrisă separat, care poate fi apelată din diferite puncte ale unui program. Mecanismul de implementare a subrutinelor este realizat cu ajutorul instrucţiunilor CALL şi RET. Subrutinele pot fi NEAR (în acelaşi segment cu programul apelant/intrasegment) sau FAR (într-un segment diferit/extrasegment). La apelarea unei subrutine (prin CALL), adresa unde urmează a se face revenirea este salvată pe stivă, iar la revenirea din rutină (prin RET) adresa este refăcută din stivă iar registrul IP (eventual şi CS) se reîncarcă. Mecanismul de apel a unei subrutine este ilustrat în figura

1. program principal
2. apel subrutină
3. execuţie subrutină
4. întoarcere din subrutină
5. reluare program principal



Fig. 10.1. Mecanismul de appel al unei subroutine

Apelurile intrasegment salvează doar offsetul adresei de revenire, iar la RET această valoare este reîncărcată în IP. Apelurile intersegment salvează şi conţinutul registrului CS şi cel al registrului IP, astfel că, la revenire, se vor reface ambele. Pentru cele două cazuri, instrucţiunile CALL şi RET au coduri diferite.Instrucţiunea RET nu are operanzi; instrucţiunea CALL are un singur operand. Modurile de adresare sunt identice cu cele de la JMP, cu excepţia adresării relative. În cazul în care subrutina alterează (foloseşte) regiştrii a căror valoare este necesară în continuare în programul ppellant, aceştiregiştri trebuie salvaţi pe stivă cu PUSH şi apoi refăcuţi cu POP înainte de revenirea din subrutină.

 ÎntreruperiÎntreruperea este un semnal transmis sistemului de calcul prin care acesta este anunţat de apariţia unui eveniment care necesită atenţie.

Atunci când evenimentul s-a produs, au loc, în ordine, următoarele acţiuni: **-** suspendarea programului în curs de desfăşurare; salvarea pe stivă a adresei de revenire (IP, CS); **-** lansarea în execuţie a unei rutine specializate numită rutină de tratare a întreruperii care deserveşte întreruperea; **-** reluarea execuţiei programului suspendat prin refacerea de pe stivă a adresei de revenire.Cauzele acestor evenimente pot fi de două tipuri: interne şi externe. O întrerupere este luată în considerare numai între execuţiile a două instrucţiuni maşină succesive. Dacă apar simultan două întreruperi, circuitele hard ale sistemului de calcul decid care dintre ele va fi servită prima.La apariţia unei întreruperi, sistemul de calcul trebuie, în această ordine: **-** să determine tipul evenimentului care a generat întreruperea (intern, extern), **-** să afle care este cauza întreruperii, **-** să determine adresa rutinei de tratare a întreruperii (RTI).Pentru fiecare tip de eveniment şi pentru fiecare cauză posibilă se construieşte câte o RTI. Metoda folosită pentru localizarea rapidă a RTI este vectorizarea întreruperilor. Aceasta constă în a asocia pentru fiecare întrerupere o locaţie de memorie cu adresă fixă. În această locaţie se trece adresa RTI corespunzătoare întreruperii. Microprocesorul 8086 dispune de 256 întreruperi numerotate de la 00h la FFh. Vectorizarea acestora se realizează astfel: la începutul memoriei RAM sunt rezervate 256 dedublucuvinte; fiecare dublucuvânt conţine o adresă FAR a unei RTI. Primul dublu cuvânt, aflat la adresa **0000:0000**, conţine adresa RTI pentru intreruperea 00h; al doilea, aflat la adresa **0000:0004** conţine adresa RTI pentru întreruperea 01h, etc. Pentru o întrerupere k, dublul cuvânt careconţine adresa RTI se află la adresa **0000:4\*k**



Întreruperile se clasifică în întreruperi hard şi soft. Întreruperea hard 00h provine de la microprocesor (internă) şi apare la tentativa de împărţire la zero; întreruperea 08h provine de la circuitul de temporizare şi este folosită pentru contorizarea timpului; întreruperea 02h semnalizează eroare de paritate la accesarea unei locaţii de memorie şi este o întrerupere nemascabilă, adică declanşarea ei nu poate fi controlată prin flagul **Interrupt.**Întreruperile soft oferă accesul la servicii BIOS şi servicii DOS; sunt foarte mult utilizate datorită facilităţii oferite, o bază de programe care poate fi folosită ca o librărie de programe (rutine) gata scrise, care uşurează mult munca programatorului în limbaj de asamblare. Aceste rutine poartă numele de servicii. Se vor exemplifica servicii pentru câteva întreruperi folosite intensiv în scrierea programelor.Mecanismul de tratare a unei întreruperi este asemănător cu cel al subrutinelor: **-** la lansarea unei întreruperi, indiferent de felul acesteia, starea curentă a microprocesorului este salvată pe stivă; se salvează pe stivă şi PSW;alte întreruperi sunt dezactivate; **-** microprocesorul identifică adresa unde se află subrutina de tratare a întreruperii; în acest scop, numărul asociat întreruperii (tipul întreruperii) este folosit ca index în tabloul vectorilor de întreruperi; **-** se încarcă în CS şi IP adresa subrutinei din poziţia corespunzătoare a tabloului vectorilor de întrerupere; se execută rutina de tratare a întreruperii până la întâlnirea instrucţiunii IRET; **-** se revine din întrerupere prin reîncărcarea lui IP, CS şi PSW cu valorile salvate pe stivă la apelare.

Instrucţiuni specifice întreruperilorO întrerupere soft poate fi apelată prin instrucţiunea INT, cu sintaxa INT n, provocând activarea handler-ului corespunzător întreruperii cu numărul n. Ea realizează patru acţiuni succesive: **-** pune în stivă flagurile (PSW); **-** pune în stivă adresa FAR de revenire (CS, IP); **-** pune 0 în flagurile TF şi IF; **-** apelează prin adresare indirectă handlerul asociat întreruperii.Instrucţiunile STI şi CLI acţionează asupra flagului de întreruperi IF, indicând procesorului cum să se comporte la apariţia unei întreruperi. După CLI (Clear Interrupt, IF=0), procesorul nu mai acceptă vreo întrerupere. Apare de obicei la începutul unui handler pentru a evita perturbarea activităţii acestuia.STI (Set Interrupt) permite procesorului să accepte întreruperi, IF=1.OBS. Întreruperile nemascabile nu ţin cont de starea flagului IF!

Instrucţiunea IRET provoacă revenirea dintr-o întrerupere. Ea este ultima instrucţiune executată într-un handler, având efect complementar instrucţiunii INT:

- reface flagurile din stivă;

- revine la instrucţiunea a cărei adresă FAR se află în vârful stivei

|  |  |  |
| --- | --- | --- |
| **INT N** SP ← Sp-2 | **IRET**IP ← [SP] |  |
| [SP] ← PSW  | SP ← SP+2 |  |
| I ← 0  | CS ← [SP] |  |
| SP ← SP-2 [SP] ← CS SP ← SP-2  | SP ← SP+2PSW ← [SP]SP ← SP+2 |  |
| [SP] ← IPIP ← [4\*N+1 4\*N]CS ← [4\*N+3 4\*N+2] |  |  |

Întreruperi şi servicii BIOS şi DOS
BIOS-ul răspunde de gestionarea echipamentelor de intrare/ieşire. Deci, în BIOS sunt scrise o serie întreagă de subrutine legate de aceste echipamente. Apelarea lor într-o aplicaţie se face prin întreruperi; în cadrul fiecărei întreruperi se pot executa mai multe servicii, selecţia serviciului dorit
fiind realizată prin încărcarea în registrul AH a unui număr specific serviciului, înainte de apelarea întreruperii. Parametrii de apel ai serviciului se încarcă în anumiţi regiştri, după caz.
Câteva întreruperi BIOS sunt prezentate mai jos:
INT 10 h - servicii de ecran / video
INT 13 h - servicii de disc
INT 14 h - servicii de comunicaţii seriale
INT 16 h - servicii de tastatură
INT 20 h - terminare program
Servicii video - INT 10 hFuncţia 00 - setarea modului video
AH = 00
AL = codul modului video (vezi Anexa 3)
Funcţia 02 - setarea poziţiei cursorului
AH = 02
BH = numărul paginii video (0 pentru modul grafic)
DH = rândul
DL = coloana
Funcţia 09 - scrierea caracterului la cursor (fără deplasarea cursorului)
AH = 09

AL = codul ASCII al caracterului de scrisBH = pagina videoBL = atribut de culoare **Funcţia 0ch** - scrierea unui pixel grafic la coordonateAH = 0chAL = culoareaBH = pagina videoCX = coloanaDX = rândul **Funcţia 0eh** - scrierea unui caracter în mod teletype (cu deplasarea cursorului)AH = 0ehAL = codul ASCII al caracterului de scrisBH = pagina video **Servicii tastatură - INT 16hFuncţia 00** - aşteaptă o tastă şi citeşte caracterul tastatAH = 00AL = (returnat) codul ASCII al caracterului tastat **Funcţia 01** - citeşte starea tastaturiiAH = 01Z = 0 - s-a apăsat o tastă; Z = 1 - nu s-a apăsat tastă **Funcţii DOS**Principala întrerupere DOS este 21h. Sarcinile unora dintre întreruperile amintite mai sus au fost preluate şi uneori extinse de către unele din funcţiile acestei întreruperi. **Funcţia 01** - citire caracter de la tastaturăAH=01AL = (returnat) caracterul citit **Funcţia 09** - scrierea unui şir de caractere terminat cu “$”AH = 09DS:DX - pointer la un şir ce se termină cu caracterul “$” **Funcţia 4ch** - terminarea procesului cu cod de returAH = 4chAL = cod de retur metodă de terminare a unui program; nu este suportată de versiuniDOS sub 2.x. **Redirectarea unei întreruperi**Rutina de tratare a unei întreruperi oarecare poate fi rescrisă de către utilizator, respectând anumite reguli, legate în primul rând de structura rutinei respective, apoi de manipularea adreselor acestor rutine. În scopul înlocuirii unei RTI cu o rutină scrisă de utlizator, se va înlocui la adresa corespunzătoare rutinei în TVI cu adresa noii rutine. Ca o măsură firească de precauţie, cel care modifică adresa unui handler trebuie să păstreze adresa veche şi să o refacă atunci când nu mai doreşte folosirea handlerului propriu. **Funcţia 35** a întreruperii 21h permite citirea adresei RTI din TVI pentru o anumită întrerupere dorită de utilizator. Se pune în AH codul funcţiei (35h) iar în AL tipul întreruperii. După apelul INT 21h în regiştrii ES:BX se obţine adresa FAR a handlerului.  **Funcţia 25h** a întreruperii 21h permite modificarea adresei RTI în TVI pentru o anumită întrerupere dorită. În AH se pune codul funcţiei (25h), în AL se pune tipul întreruperii dorite, iar în DS:DX se pune adresa FAR a noului handler.

Analiza tabelului de cod ASCII arată următoarele.

În sistemul de codificare ASCII, pentru orice cifră de la 0 la 9, raportul

 cod (cifră) - cod ('0') = cifră

Deoarece codul de caractere 0 („0”) este 30h, codul ASCII al oricărei cifre de la 0 la 9 diferă de reprezentarea binară corespunzătoare a numărului cu 30h.

Prin urmare, pentru a converti codul de caractere ASCII („0” .. „9”) într-un număr, scădeți 30 de ore din codul de caractere.

Când introduceți de la tastatură, de exemplu, numărul 5, nu este suficient să utilizați funcția 1h de întrerupere 21h, deoarece registrul al va conține codul de caractere „5”, și nu numărul 5.

Pentru a obține numărul 5, trebuie să scădeți și 30 de ore din codul rezultat:

 mov ah, 01h

 mov 21h; în toate codul de caractere este „5”

 sub al, 30h; acum în numărul 5

PROBLEMA 1

Să fie stocat un număr zecimal nesemnat (de la 0 la 255) în segmentul de date sub numele simbolic N. Este necesar să scrieți cifrele adresei KOD (ca simboluri) din notația zecimală a numărului.

DECIZIE

Fie N = abc, unde a, b, c sunt cifre zecimale ale lui N.

Pentru a obține cifra corectă c, trebuie să luați restul din împărțirea lui N la 10. Cocientul incomplet al divizării este numărul ab, dacă îl împărțiți la 10, atunci coeficientul incomplet va da cifra a, iar restul va da cifra b.

Pentru a obține aceste numere ca simboluri (acestea pot fi apoi afișate pe ecran), adăugați codul „0” (30h) la număr.

N db?

 KOD db 3 dup (?)

 ... ... ...

 mov bl, 10

 mov al, N

 mov ah, 0; extinderea N în ax la un cuvânt (nesemnat)

 div bl; ah = c, al = ab

 adaugă ah, 30h

 mov KOD + 2, ah; a notat ultima cifră

 mov ah, 0; al = ab, extindeți ab la cuvânt (nesemnat)

 div bl; ah = b, al = a

 add ax, '00 '; ah = b + '0', al = a + '0'

 mov KOD + 1, ah; a înregistrat media

 mov KOD, al; a notat prima cifră

PROBLEMA 2

Introduceți un număr zecimal și scrieți-l în segmentul de date sub numele simbolic N (un octet în mărime).

DECIZIE

La rândul său, introduceți numerele și formați numărul conform schemei lui Horner.

Să fie introduse deja primele cifre ale numărului, de exemplu, 2 și 1, iar din ele se formează numărul 21. Să fie introdus următorul număr 3. Apoi înmulțim numărul anterior cu 10 și adăugăm o nouă cifră la acesta: 21 \* 10 + 3 = 213. Și așa pentru fiecare cifră nouă.

Cometariu. Vom presupune că utilizatorul introduce corect un număr în intervalul 0-255 și că intrarea este finalizată apăsând tasta Enter.

N db?

 ... ... ...

 ah, 01h

 int 21h; în al - primul simbol

 sub al, 30h; acum prima cifră

 mov ah, 0; extensie la cuvânt

 mov bx, 10

 mov cx, topor; în cx - prima cifră

Buclă: ah, 01h

 int 21h; următorul simbol

 cmp al, 0dh; comparație cu caracterul Enter

 je End; sfârșitul intrării

 sub al, 30h; în al - cifra următoare

 cbw; extensie la cuvânt

 xchg topor, cx; acum ax este numărul anterior, cx este următorul

 mul bx; ax \* 10

 add cx, ax; cx = ax \* 10 + cx

 jmp Loop; continuarea intrării

 End: mov N, cx

**Lucrarea de laborator nr.4**

**De elaborat un program care citeste un numar de la tastatura in diapazonu 0-255 din numarul dat scade cifra 53 si afiseazape ecran valoarea primita**